REMARKS

Claims 1-30, 37-39 are pending in this application. Applicant urges that all of the pending claims are in condition for allowance. Applicant respectfully requests reconsideration of the outstanding rejections and allowance of all pending claims in view of the reasons set forth below.

I. Claim Rejections

In the Office Action, claims 1-11, 13, 16-26, 28, and 39 were rejected under 35 U.S.C. § 102(b) as anticipated by Conway ("Parsing with C++ deferred expressions", ACM SIGPLAN Notices, vol. 29, no. 9, pp. 9-16, ACM, 1994), hereafter "Conway"). (Office Action at page 2).

In the Office Action, claims 12, 14, 15, 27, 29, 30 and 37-38 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Conway. (Office Action at page 7).

II. Claim Rejection under 35 U.S.C. § 102(b)

In the Office Action, claims 1-11, 13, 16-26, 28 and 39 were rejected under 35 U.S.C. § 102(b) as anticipated by Conway. Applicant respectfully traverses the rejection.

A. Claim 1

Claim 1 recites:

> In a program development environment, a method comprising the steps of:
> providing, via a programming language, ***a language processor with built-in support for a parse tree data structure*** written in a base language, said parse tree data structure represented as a class, said class being the basis for a plurality of parse tree objects, said parse tree objects including methods that retrieve values for base language objects;
> defining an assignment function, said assignment function taking a plurality of parse tree structures as arguments; and
> calling said assignment function to determine the value of at least one assignment within at least one of said base language and a base language extension to said base language.

Claim 1 requires ***a language processor with built-in support for a parse tree data structure***. "Built-in support" means that the parse tree data structure is implemented in the language processor without reference to any external code. Conway does not disclose at least ***a language processor with built-in support for a parse tree data structure.*** In particular, as will be described below, Conway does not disclose ***built-in support***.

1. The Examiner's Reiterated Position

The Examiner suggests that ***a language processor with built-in support for a parse tree data structure*** is present in Conway at paragraph 2 of "The Deferred Expression Idiom" section on page 9 (Office Action at page 2), which recites:

> In a similar manner, the classes described in this paper used overloaded operators (in this case, virtually all of the operators in the C++ language) to generate an object representing the parse tree of the expression. ***This expression object belongs to a class derived from the root*** **Parser** ***class, and hence the original expressions used to generate it may be incorporated seamlessly into any embedded grammar built with the*** **Parser** ***library***. Figure 1 illustrates this technique, components of which are described in Sections 3 through 8.

Conway is generally directed to extending a C++ class. (Conway at Abstract). Conway takes the preexisting *Parser* class library (Conway at page 9, first paragraph, lines 1-2) and preexisting classes developed in a publication by Van Wyk (Conway at page 9, section 2, first two paragraphs), and further extends Van Wyk's classes. (Conway at Abstract). This extension allows parsing actions to be specified directly as part of the grammar, rather than indirectly as function calls. (Conway at Abstract; Conway at page 9, paragraph 4). To implement this extension, Conway overloads common operators so that they generate an object which represents their respective operations. (Conway at page 9, section 2, paragraph 1).

Applicant respectfully disagrees with the Examiner's interpretation of paragraph 2 of Conway, because the root Parser class is not built into Conway. Conway states that the objects he uses derive from the root Parser class; the root Parser class is a separately implemented library that is not "built-in."

2. The Examiner's Response to Applicant's Arguments

At page 8 of the present Office Action, the Examiner responds to the argument that Conway lacks built-in support, as Applicant asserted in the amendment of November 2, 2007, as follows:

> [T]he parser class library defines a set of related classes to build parsing grammars within a C++ program (page 9, first paragraph, lines 1-2) and the C++ parse expression objects for operator overloading are built from the library source code in C++ with a C++ language processor with built-in support for such a parse expression structure to be generated (sixth paragraph lines 1-5).
>
> Applicant respectfully disagrees.

a. Page 9, first paragraph, lines 1-2

Applicant respectfully points out that page 9, first paragraph, lines 1-2 does not disclose what the Examiner believes these portions of Conway disclose. Page 9, first paragraph, lines 1-2 states "the *Parser* class library defines a set of related classes which may be used to build parsing grammars within a C++ program."

The first paragraph of Conway does not discuss ***built-in support for a parse tree data structure.*** In fact, at page 9, first paragraph, Conway explicitly states that the class library relies on to build his parsing grammars is defined in an ***external*** library. While Conway may implement function definitions for members of the *ParserExpr* class, this constitutes only half of the process of creating a function. In order to utilize a function, it must be both declared and defined. The step of declaring a function is conventionally handled in a header (i.e., a ".h") file. The necessary step of declaring the functions is not built-in to Conway, because it is handled by the ***external*** "Parser.h" library.

b. Page 9, sixth paragraph, lines 1-5

The Examiner further elaborates at page 8 of the Office Action, stating "the C++ parse expression objects for operator overloading are built from the library source code in C++ with a C++ language processor with built-in support for such a parse expression structure to be generated." The Examiner suggests that support for this statement can be found in Conway at page 9, paragraph 6, lines 1-5. (Office Action at 8). The sixth paragraph on page 9 equates to

page 9, section 2 ('The Deferred Expression Idiom'), paragraph 2. This portion of the reference has been discussed in detail above in section 1, "The Examiner's Rejection." As noted above, this portion does not disclose ***built-in support for a parse tree data structure***

c. Further evidence that Conway does not disclose *built-in support for a parse tree data structure*

Figure 1 further evidences that Conway does not provide ***built-in support for a parse tree data structure***. Conway describes that "all classes representing deferred expressions are derived from a common base class, *ParserExpr*, which is in turn directly derived from the root class *Parser*." (Conway at page 10, section 3, paragraph 1). As can be seen in Figure 1, the header file for the root class *Parser* is imported into the code through a "#include 'Parser.h'" instruction. Thus, Figure 1 shows that Conway relies on external libraries and not built-in support for a parse tree data structure.

There may be several advantages to the built-in support of claim 1. For example, efficiency may be improved, because the system need not look for external header files. Further, the method can be practiced in program development environments that are not object-oriented, as described in the Specification at page 6, paragraph 1. This is an advantage which the system of Conway does not share. Moreover, the built-in support may allow the source code to be compiled faster, requires less storage space for the program, and allows the code to be compiled even on systems which do not have a copy of the requisite header files.

For at least these reasons, Applicant urges that claim 1 is in condition for allowance and requests that the Examiner withdraw the 35 U.S.C. §102(b) rejection of claim 1.

B. Claims 2-11 and 13

Claims 2-11 and 13 depend from claim 1 and thus include all of the elements of claim 1. Therefore, Applicant submits that claims 2-11 and 13 are also in condition for allowance, and requests that the Examiner withdraw the 35 U.S.C. §102(b) rejection of claims 2-11 and 13.

Further, claim 5 recites ***evaluating the class at compile time, and adjusting the resulting class definitions from said evaluation to increase the efficiency of run-time performance***. The Examiner suggests that Conway discloses this feature of claim 5 at page 10, lines 3-6. Conway at page 10, lines 3-6 recites:

> Terminals and rules within the grammar evaluate trivially (that is, their *Evaluate()* methods do nothing) whilst objects of class *Action* call the function associated with them. Calling the *Evaluate()* method of one of the new expression objects causes the entire expression tree contained in it to be evaluated, effectively executing the original expression embedded in the grammar.

This section of Conway does not discuss evaluating ***the class*** at compile time. It describes evaluating "terminals and rules within the grammar." Terminals and rules are not the same as ***the class***, which is a representation of the parse tree data structure, and is the basis for a plurality of parse tree objects. (Application at Claim 2). Moreover, this section of Conway does not discuss evaluating the class ***at compile time***. Rather, the cited portion of Conway appears to be discussing evaluation at ***run-time***.

Further, there is no discussion of ***adjusting the resulting class definitions from said evaluation to increase the efficiency of run-time performance***. No adjustment to the class definitions is made from the evaluation. Conway is also silent concerning the efficiency of run-time performance.

For at least the reasons described above, Applicant submits that claim 5 is in condition for allowance, and requests that the Examiner withdraw the 35 U.S.C. §102(b) rejection of claim 5.

C. Claims 16-26, 28 and 39

Independent claims 16 and 39 also recite ***a language processor with built-in support for a parse tree data structure*** and are therefore allowable for the same reasons set forth above with regard to claim 1. Further, claim 20 recites the same elements as claim 5, which further define

patentable differences over the cited prior art, as discussed above. Therefore, Applicant requests that the Examiner withdraw the 35 U.S.C. §102(b) rejection of claim 16.

Claims 17-26 and 28 depend from claim 16 and thus include all of the elements of claim 16. Therefore, Applicant requests that claims 17-26 and 28 are also in condition for allowance and requests that the Examiner withdraw the 35 U.S.C. §102(b) rejection of claims 17-26 and 28.

III. Claim Rejection under 35 U.S.C. § 103(a)

The Examiner rejects claims 12, 14, 15, 27, 29, 30 and 37-38 under 35 U.S.C. § 103(a) as being obvious over Conway. The Examiner stated that, while Conway did not disclose the limitations of claim 12 (an assignment function used to perform multiply and accumulate (MAC) operations), and claims 29 and 30 (using parse tree classes for an embedded processor and processor emulation), the limitations would have been obvious modifications to the Conway system. (Office Action at page 7). The Examiner rejects claims 14 and 15 as corresponding to claims 29 and 30, respectively. The Examiner also rejects claim 27 as corresponding to claim 12 and rejects claims 37 and 38 for the same reasons as he rejected claims 14 and 15. Applicant respectfully traverses the rejection.

Claims 14-15 and 37-38 are independent claims and recite ***a language processor with (having) built-in support for a parse tree data structure***. As discussed above in connection with claim 1, this claim element is not disclosed by Conway. This feature is further not suggested by Conway, which is directed to the design of external class parser libraries. In fact, Conway suggests an approach that is not built-in, but explicitly made to depend on external libraries. Therefore, Conway appears to teach away from ***a language processor with built-in support for a parse tree data structure***. Accordingly, Conway does not support a valid 35 U.S.C. §103 rejection of claims 14-15 and 37-38. Applicant requests that the Examiner withdraw the 35 U.S.C. §103(a) rejection of claims 14-15 and 37-38.

Claims 12, 27 and 29-30 are dependent claims which depend from independent claims reciting ***a language processor with built-in support for a parse tree data structure***. Claims 12,

27 and 29-30 are therefore also in condition for allowance for the same reasons set forth above with respect to claims 14-15 and 37-38. Applicant therefore respectfully requests that the Examiner withdraw the 35 U.S.C. §103(a) rejection of claims 12, 27 and 29-30.

CONCLUSION

In view of the above Remarks, Applicant believes the pending application is in condition for allowance and urges the Examiner to pass all of the pending claims to allowance. Should the Examiner feel that a teleconference would expedite the prosecution of this application, the Examiner is urged to contact the Applicant's attorney at (617) 227-7400.

Please charge any shortage or credit any overpayment of fees to our Deposit Account No. 12-0080, under Order No. MWS-039. In the event that a petition for an extension of time is required to be submitted herewith, and the requisite petition does not accompany this response, the undersigned hereby petitions under 37 C.F.R. §1.136(a) for an extension of time for as many months as are required to render this submission timely. Any fee due is authorized to be charged to the aforementioned Deposit Account.

REMARKS

In view of the above amendment, applicant believes the pending application is in condition for allowance.

Dated: March 17, 2008

Respectfully submitted,

By: Electronic Signature for /Kevin J. Canning/
Kevin J. Canning
Registration No.: 35,470
LAHIVE & COCKFIELD, LLP
One Post Office Square
Boston, Massachusetts 02109-2127
(617) 227-7400
(617) 742-4214 (Fax)
Attorney/Agent For Applicant